Shgrid Documentation

INTRODUCTION TO NGMATH

The ngmath library is a collection of interpolators and approximators for one−dimensional, two−dimensional and three−dimensional data. The packages, which were obtained from NCAR, are:

natgrid −− a two−dimensional random data interpolation package based on Dave Watson's nngridr.

dsgrid −− a three−dimensional random data interpolator based on a simple inverse distance weighting algorithm.

fitgrid −− an interpolation package for one−dimensional and two−dimensional gridded data based on Alan Cline's Fitpack. Fitpack uses splines under tension to interpolate in one and two dimensions.

csagrid −− an approximation package for one−dimensional, two−dimensional and three−dimensional random data based on David Fulker's Splpack. csagrid uses cubic splines to calculate its approximation function.

cssgrid −− an interpolation package for random data on the surface of a sphere based on the work of Robert Renka. cssgrid uses cubic splines to calculate its interpolation function.

shgrid −− an interpolation package for random data in 3−space based on the work of Robert Renka. shgrid uses a modified Shepard's algorithm to calculate its interpolation function.

COMPARISION OF NGMATH PACKAGES

Three−dimensional packages −− shgrid, csagrid and dsgrid.

shgrid is probably the package of choice for interpolation. It uses a least squares fit of biquadratics to construct its interpolation function. The interpolation function will pass through the original data points.

csagrid uses a least squares fit of cubic splines to calculate its approximation function: the calculated surface will not necesarily pass through the original data points. The algorithm can become unstable in data sparse regions.

dsgrid uses a weighted average algorithm and is stable in all cases, but the resultant interpolation is not usually smooth and execution time is very slow. dsgrid is probably best used when csagrid and shgrid fail or for comparative purposes.

Two−dimensional packages −− natgrid, fitgrid, csagrid and dsgrid.

natgrid is the package of choice in most cases. It implements a very stable algorithm and has parameters for adjusting the smoothness of the output surface.

fitgrid offers user−settable parameters for specifiying derivatives along the boundary of the output grid which are not available in natgrid.

csagrid produces an approximate two−dimensional surface which may be smoother than that produced by fitgrid
and natgrid.

dsgrid is not recommended for two−dimensional surfaces. natgrid is superior in all respects.

One−dimensional packages −− fitgrid and csagrid.

fitgrid is definitely the package of choice. It has many features not available in csagrid, such as interpolating parametric curves, finding integrals, handling periodic functions, allowing smoothing that varies from linear to a full cubic spline interpolation and specifying slopes at the end points.

Interpolation on a sphere −− cssgrid.

cssgrid is designed specifically for interpolating on a sphere. It uses cubic splines to calculate an interpolation function.

SHGRID PACKAGE

shgrid interpolates random data in 3−space. and it provides the means for finding the nearest point or points to a given point in 3−space. shgrid constructs a once−continuously differentiable function that interpolates the original data. This function is constructed as follows. First, for each input point a bivariate quadratic is computed that interpolates the functional value at the data point in a least squares fit to neighboring data values. How close the bivariate quadratic comes to the functional values is attenuated by an inverse distance weighting. That is, the calculated bivariate quadratic will fit the functional values at nearby points better than at farther points. After the bivariate quadratics are calculated, the interpolating function is calculated as a weighted average of the bivariate quadratics.

SHGRID CONTENTS

Access through Python to the shgrid package from NCAR's ngmath distribution is provided directly through the module
shgridmodule.so which was generated by connecting the Fortran routines to Python using the Pyfort implementation
by Paul Dubois.

Single precision procedures:

shgrid −− interpolates 3D random data

shgetnp −− finds nearest points to a given point in 3−space

shseti −− sets values for the integer control parameters

shgeti −− retrieves values for the integer control parameters

REQUIRED FILE

shgridmodule.so −− the Python interface to the ngmath shgrid package.

USEFUL FILES

sh.py −− the object oriented interface including a general help package.
shgridtest.py −− the code to test sh.py and to write documentation.

USAGE

This module is designed to use in two ways. One is through the use of the object oriented interface to the underlying
functions. This approach is recommended for users not already familiar with the original shgrid distribution because
it simplifies the calls to the routines. The other method uses the original functions calling them directly from Python.

−−
−−−−−−−−−−−−−−−−−− OBJECT ORIENTED APPROACH −−−−−−−−−−−−−−−−

The sh module contains the Shgrid class and the single method rgrd. Use of this object oriented scheme is a two
step process. The first is making an instance of the class which envolves submitting the appropriate grid for the
task at hand. There are two choices: make an interpolation to a three dimensional grid or find the nearest points
to a specific point which has been already set in making the instance.

STEP 1.

Array Input And Output Case In Preparation For Interpolation In 3−space

To make an instance, r, type:

import sh

r = sh.Shgrid(xi, yi, zi, xo, yo, zo)

where xi, yi and zi are the input coordinate arrays in list format
while xo, yo and zo are the output grid coordinate arrays which
must be increasing but need not be evenly spaced.

Array Input And Single Point Output Case In Preparation For Finding Nearest Points In 3−space

To make an instance, r, type:

import sh

r = sh.Shgrid(xi, yi, zi, px, py, pz)

where xi, yi and zi are the input coordinate arrays in list format
while px, py and pz are the coordinates of a point whose nearest
neighbors are to be found.

To look at the default settings for the control parameters and a brief description or thier
properties, type

r.printDefaultParameterTable()

To change a setting type the new value. For example, to set ncl to 8, type

r.ncl = 1

To find an individual value, type the name. For example, to exam the value of nfl, type

r.nfl

To check the settings type

r.printInstanceParameterTable() −− prints the table with values and a description of the parameters used in subsequent calls to the method function rgrd
or

r.printInstanceParameters() −− prints a list of the parameters values used in subsequent calls to the the rgrd method

sh. printStoredParameters() −− prints the parameters in memory which may differ from the above if the user has made more than one instance of the Shgrid class.

In the absence of an instance of the class Shgrid, a description of the control parameters can be found by typing

import sh
sh.printParameterTable()

STEP 2.

Interpolation In 3−space

Type

dataOut = r.rgrd(dataIn)

where dataIn is input data in list form and dataOut is output data in gridded form

Finding Nearest Points In 3−space

Type

np = r.rgrd(numberPoints)

where:
np is an array with the indices into the input data arrays of the sequence of nearest points
numberPoints is the number of nearest points requested. It may be 1.

For the ith element in np the point in 3−space is (xi[np[i]], yi[np[i]], zi[np[i]]).

−−
−−−−−−−−−−−−−−−−−−−−− ORIGINAL FUNCTION APPROACH −−−−−−−−−−−−−−−−−−−−

The module shgridmodule.so exports the following functions to Python from the original Fortran library:

Single precision procedures:

shgrid −− interpolates 3D random data

shgetnp −− finds nearest points to a given point in 3−space

shseti −− sets values for the integer control parameters

shgeti −− retrieves values for the integer control parameters

Information on the use of the routines is available by importing shgridmodule and printing the docstring of interest. For example, documentation for the routine shgrid is obtained by typing

import shgridmodule

print shgridmodule.shgrid.__doc__

The documentation associated with the shgridmodule.so, such as the doctrings, describe the Fortran code. The indices start with 1 (not 0 as in the Python interface) and the arrays are transposes as nomenclature. For example, the Fortran array dimensioned as (x,y,z) is the same physically as the Python or C array dimensioned as
(z,y,x). Since the calls are from Python the array indices start with 0. It is only in a request for a specific index submitted to Fortran that the indices start at 1.

The long and the short of this is as follows. If you use the interface, indices start at 0. If you call the routines directly, in specific requests indices start at 1. In either case, data is passed with the x index moving fastest.

This same information is available in the help package.

import sh

print sh.help('shgrid')

A description of the control parameters is not in the shgridmodule documentation. It can be found by typing

import sh
sh.printParameterTable()

DOCUMENTATION

Documentation is provided through Python's docstrings, essentially Python style program comments. A help package
provides instructions on the use of shgrid. A table of contents is printed to the screen by typing

sh.help()

after importing sh.

A hard copy of this documentation is written to the file shgridmodule.doc after import shgridtest by typing

shgridtest.document()

TESTING

To run a some tests of the 3D interpolation and the nearest point computattion and to get a copy of this documentation,
type

cdat shgridtest.py
